Boston University Computer Science Technical Report No. 99-007, July 1999.

Combinations of Deformable Shape Prototypes

Saratendu Sethi and Stan Sclaroff
Image and Video Computing Group
Computer Science Department
Boston University, Boston, MA 02215

Abstract

We present a model-based technique for encoding non-rigid object classes in terms of object prototypes. Objects from the same class can be parameterized by identifying shape and appearance invariants of the class to devise low-level representations. The approach presented here creates a flexible model for an object class from a set of prototypes. This model is then used to estimate the parameters of low-level representation of novel objects as combinations of the prototype parameters. Variations in the object shape are modeled as non-rigid deformations. Appearance variations are modeled as intensity variations. In the training phase, the system is presented with several example prototype images. These prototype images are registered to a reference image by a finite element-based technique called Active Blobs. The deformations of the finite element model to register a prototype image with the reference image provide the shape description or shape vector for the prototype. The shape vector for each prototype, is then used to warp the prototype image onto the reference image and obtain the corresponding texture vector. The prototype texture vectors, being warped onto the same reference image have a pixel by pixel correspondence with each other and hence are "shape normalized". Given sufficient number of prototypes that exhibit appropriate in-class variations, the shape and the texture vectors define a linear prototype subspace that spans the object class. Each prototype is a vector in this subspace. The matching phase involves the estimation of a set of combination parameters for synthesis of the novel object by combining the prototype shape and texture vectors. The strengths of this technique lie in the combined estimation of both shape and appearance parameters. This is in contrast with the previous approaches where shape and appearance parameters were estimated separately.

1 Introduction

One of the primary goals of *computer vision* is to optimally and reliably compute object descriptions from images. Such descriptions are useful for various purposes like object recognition and analysis. Important characteristics of such descriptions are that they should be easily computable and unique. Various methods have been devised in the past that recover object descriptions from images for recognition. A survey of such techniques focussed on face recognition methods reported by Fromherz and others[12] implies that most recognition systems have been demonstrated on datasets taken under constrained conditions that tend to breakdown under varying pose and lighting conditions. Statistical methods, which have the potential to estimate these variations, suffer from the problem of requiring large amounts of training data[10]. We intend to formulate a technique that will perform reliably with such varying conditions and at the same time will estimate a flexible model of the object class from a small set of prototypes.

A common strategy employed in computer vision to design effective algorithms is to emulate methods of reasoning believed to be used by human beings to perform *image analysis*. The following subsection summarizes some of the findings made in the community of psychophysics. Based on these observations, we lay our framework, the objective of which is to simulate the observations by using existing computer vision techniques.

1.1 Human Visual System

Various psychophysical and physiological studies [11, 40, 23, 24, 4, 29] have indicated that the human visual system uses strategies that represent objects in 2D rather than 3D models. Several psychophysical tests have been conducted that explore different aspects of the problem of recognition and representation in human vision. In a nutshell, the following conclusions have been made from the outcomes of a variety of psychophysical tests[4, 23]:

- *Multiple views*
  Evidence from various tests indicate that humans encode three dimensional objects as multiple viewpoint-specific representations that are largely two-dimensional.
- *Normalization*
  Various stimulus tests have indicated that subordinate level recognition is achieved by employing a time consuming normalization process to match objects seen in unfamiliar viewpoints to familiar stored viewpoints.
- *View Interpolation*
  Various test evidences and computational simulations indicate that view interpolation offers a plausible explanation for viewpoint dependent performance of human response times and error rates for recognition.

These results imply that the human visual system probably uses heuristics-based techniques which rely on multiple views representation in terms of 2D rather than explicit

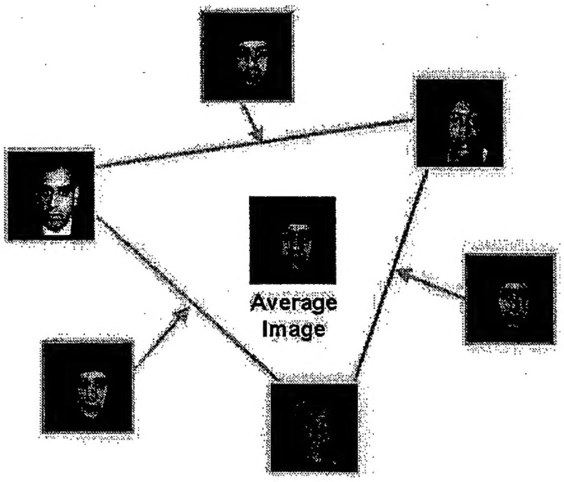

Figure 1: Basic Idea: The images at the vertices of the triangle represent three prototypes. The prototypes can be registered with each other by warping them onto the average image. The shape and texture of each prototype can then be combined to generate new images. Here the intermediate images are represented along the edges of the triangle. For each intermediate image, the contribution of the adjacent prototypes is more than that of the one farther away.

3D models with appropriate depth information. This viewpoint is further supported by recent advances in computational theories based on view interpolation[11].

1.2 Basic Idea

The abovementioned results strongly indicate that the human mind stores a set of prototypes with significant variability and uses their combinations to recognize an object. This motivates us to design algorithms that will represent object classes in terms of 2D prototype images. As a starting point, it will be worthwhile to study some existing computer vision models and extend them to simulate the human visual system. Intuitively, an object can be thought of as comprised of two components: shape and appearance[1]. Based on this premise, we can categorize current computer vision models broadly into two classes, namely *shape models*, that account for the object morphism and *appearance models*, that account for the object physiognomy. Objects from the same class can be defined in terms of small variations from an average object. These variations are defined as displacements in an orthogonal coordinate system where each axis represents a principal direction of variation, learned statistically from a large set of examples.

Assuming that an object can be parameterized in terms of its shape and texture features, we define our goal as follows. "Given sufficient number of good prototypes that encompass appropriate in-class variations, we intend to build a deformable appearance model that will reliably describe the object class and explore the feasibility of modeling the object space by a linear combinations approach". The parameters of a novel object will be expressed as a linear combination of the parameters of the prototype objects in the training set. Figure 1 gives a pictorial description of the approach. The technique for linear combinations is exhibited for three prototype images.

1.2.1 Existence of coupling between shape and appearance parameters

An important observation made from most of the existing shape and appearance models is that either approaches parameterize only one feature and ignores the other. For instance, shape models account for shape deformations only and ignore texture parameterization. Their objective is to estimate the parameters of a warping function that would warp the template image such that it matches the novel image as closely as possible. Similarly, appearance models ignore shape parameterization by presuming that the input images are shape normalized or have a dense correspondence defined with respect to each other. Principal directions of texture variations are learned from the example images using eigen-based techniques. Some recent approaches proposed by Cootes[6] and Nastar[28] have shown that shape and appearance can be modeled together by employing the eigen-based techniques in the combined feature space. Although, both methods have been shown to work effectively for various object classes, it cannot be inferred that the underlying linearity assumption of the combined shape and texture space can be generalized to all object classes. Hence we estimate the shape and the texture parameters using non-linear optimization techniques without any assumption about linearity. This issue will be further analyzed during the preliminary tests of our system.

In Section 2 we introduce various shape and appearance models previously reported in the computer vision community. This section also discusses some of the recent methods that have proposed combined shape and appearance parameterization. The mathematical formulation of the deformable model is presented in Section 3. This section develops the combination of prototypes paradigm and derives the objective function. Section 4 describes the modeling of non-rigid deformable objects by *finite element based* methods. The section gives a brief description of *Active Blobs* which will be used for registration of objects. Section 5 describes the various minimization techniques that will be used for registration and model fitting later. Section 6 describes the implementation of various phases of the system in detail and finally summarizes the algorithm. Preliminary results are reported in Section 7. This is followed by the discussion of issues that explain the observed results in Section 8. Finally we conclude with discussion for future applications of the methodology in Section 9.

[1] The notion of appearance is same as the object texture. Both terms will be used interchangeably

2 Background and Previous Work

This section throws light on some of the deformable shape and view-based representations relevant for the formulation of our problem and then explores some methods that try to combine those representations to come up with a model that makes the paradigm of linear combinations feasible.

2.1 Shape Models

Initial shape representation methods concentrated on ways to employ flexible models by constraining the solution space of allowed deformations. Kass, Witkin and Terzopoulos[20] described a method of representing objects in images as *active contours* or energy minimizing splines that were guided by external constraint forces and influenced by image forces along the image gradients. Cootes proposed the *Chord Length Distribution* or the *Point Distribution Model*[5], a method of shape representation that estimates the chord-lengths where each object is represented as an n-vertex polygon. The n-vertex polygons are created by placing points over the object boundaries either manually or semi-automatically. Training is done by estimating the covariances of chord lengths between each pair of vertices in the polygons created. Later Baumberg and Hogg [2] used this idea to track contours by associating points with B-splines rather than edges of a polygon. This method is dependent on the initial placement of the points of the polygon and due to this reason models only global deformations.

Sclaroff and Pentland [32, 38] had proposed a method of representing objects in terms of *modal descriptions*, which is based on the idea of describing objects by their generalized symmetries, as defined by the object's deformation modes. Unlike the *point distribution model*, which statistically modeled object shapes, this method physically modeled objects by determining the modes of free vibrations of the object. The modes of an object define an orthogonal object-centered coordinate system where each feature point can be uniquely described as a combination of those modes. This technique has been used to define deformable shape prototypes that could be used for shape-based database search[36] and track deformable objects[37].

Cootes and Taylor[7] later combined the statistically based *point distribution model* and the physically based *finite element model* for more reliable modeling of flexible objects. While the point distribution model accounted for variation across object shapes, the finite element model accounted for the vibrational modes of a single object shape. This combined formulation was shown to perform better than either of the mentioned methods alone. The major drawback of this system is the requirement of large training sets. It may be the case that some prototypes are more similar and hence may form clusters in the prototype subspace. Absence of sufficient training data will then bias the system to interpret all the prototypes as similar to the prototypes of the largest cluster in the training data. This phenomenon is called *true-shape vulnerability*[15]. The point distribution models addressed this problem by using an hierarchical representation of mean/reference shapes where at each level, two nearest mean shapes were combined to obtain the mean shape in the next higher level. The hierarchy was built in a bottom-up fashion starting with all the prototypes at the lowest level.

2.2 Appearance-based Models

Appearance-based models seek to obtain a compact representation for intensity distribution. One such set of techniques employ eigen-based methods to compress an image by projecting it onto a low-dimensional orthogonal basis, the *eigenspace*[43]. The eigenspace is constructed as follows. Given a sufficient number of prototype images expressed as column vectors in the object class matrix, we use *principal components analysis* (PCA) or Karhunen-Loeve expansion to estimate the intensity distribution. The eigenvectors of the covariance matrix span the variations across the object space as captured by the prototypes. The eigenvectors corresponding to the higher eigenvalues define the directions of maximum variations and hence are chosen to represent the eigenspace. Compact representation for a new image is obtained by just projecting the image onto the eigenvectors. [27, 26, 31, 43] have shown the effectiveness of the said method especially for visual recognition. The main weakness of eigen-based techniques is that they are not robust to shape variations across the prototypes as they require the prototypes to be registered with each other.

Covell[9] has proposed a similar method based on the *principal component analysis* to define correspondences between faces in terms of feature point locations. Feature correspondences so obtained were then used for motion estimation and morphing[8] between different frames of video. This is an interesting idea where feature points are automatically placed at correct locations. The concept of point distribution models was extended to model intensity distributions. These models are known as *Appearance models*[21] and have been claimed to address the problem of shape normalization which was not addressed in eigenfaces. This method requires labeled examples for training.

2.3 Combined Parametric Models

In order to avoid the implicit parameterization of shape in appearance models and make shape models more photorealistic, there has been a growing interest in modeling both shape and appearance in a single model. Nastar, Moghaddam and Pentland[28] had combined physically based modes of vibrations with statistically-based modes of variation by considering each point in the image as a triplet of $(x, y, I(x, y))$ and doing manifold matching in this XYI space. Although this method combined both the statistical and physical modes of variation, it is dependent on good initialization.

Ullman and Basri[44] have showed that an object can be represented as a combination of 2-D images where the images are represented in terms of some linear transformations in the 3-D space. However, this method assumes a linear framework for object deformations and handles only limited non-rigid deformations.

Poggio, Jones and Vetter[17, 18, 19, 46] have suggested that given sufficient number of prototypes, the parameter vectors define a linear space and span the model space. Any novel object can then be expressed as some combination of those prototype vectors. This method combines shape or geometry with texture or appearance in a way that minimizes both shape and appearance parameters to fit the model. This is a robust method as the problem of model fitting is solved as a global non-linear minimization problem.

Cootes, Edwards and Taylor[6] have also suggested a combined formulation of their appearance and active shape models to develop a new model known as the *Active Appearance Models.* This method does PCA in both the shape and the texture spaces separately and then combines them and again does PCA to remove redundancies between shape and texture parameters. All objects are then represented as some combination in this orthogonal model.

3 Mathematical Formulation

Let $I_1, I_2, \ldots, I_N$ be the N prototypes available for training the system. Let I_{ref} be the reference image. The objective is to define a framework whereby all the prototype images can be combined to generate images of novel objects from the same class. The formulation described here is similar in flavor to that developed by Jones and Poggio[17], though the shape deformations are determined by finite element methods as opposed to optical flow methods. The prototype images are initially not in correspondence and hence cannot be combined. This emphasizes the determination of pixel to pixel correspondences amongst the prototype images. Let $S_1, S_2, \ldots, S_N$ be a set of shape parameters such that each S_i can be used to warp the *ith* prototype image onto the reference image, thereby bringing the prototype image into correspondence with the reference image, i.e.

$$S_i(x,y) = (\hat{x}, \hat{y}) \tag{1}$$

where $(\hat{x}, \hat{y})$ is the point in I_i which corresponds to (x, y) in I_{ref}. We define,

$$T_i(x,y) = \mathcal{W}^{-1}(I_i, S_i)(x,y) \tag{2}$$

where $\mathcal{W}$ is the warping function. Thus, for each prototype I_i in the training set, we obtain a shape vector S_i and a inverse warped texture vector T_i. Note that the texture vectors are shape-free as all of the prototype images are inverse warped onto the same reference prototype image.

Given a large number of prototypes which appropriately vary from each other with respect to different characteristics of the object class, we can define a set of parameters $\mathbf{b} = [b_1, b_2, \ldots, b_N]$ and $\mathbf{c} = [c_1, c_2, \ldots, c_N]$ such that the shape and the texture of a novel object I_{novel} (not in the prototype set) can be derived as a combination of the prototype shape and texture parameters.

$$S_{novel} = \sum_{i=1}^{N} c_i S_i = \mathbf{c} \cdot \mathbf{S} \tag{3}$$

$$T_{novel} = \sum_{i=1}^{N} b_i T_i = \mathbf{b} \cdot \mathbf{T} \tag{4}$$

Therefore, the equation for the novel image can be defined as follows:

$$\mathcal{W}^{-1}(I_{novel}, \mathbf{c} \cdot \mathbf{S}) = \mathbf{b} \cdot \mathbf{T} \tag{5}$$

Hence the matching phase reduces to matching the the novel image, which can be done by minimizing the sum of squared differences (SSD) error

$$E(\mathbf{c}, \mathbf{b}) = \frac{1}{2} \sum_{x,y} [\mathcal{W}^{-1}(I_{novel}, \mathbf{c} \cdot \mathbf{S})(x,y) - (\mathbf{b} \cdot \mathbf{T})(x,y)]^2 \tag{6}$$

The values of the parameters $\mathbf{c}$ and $\mathbf{b}$ so obtained, provide a compact representation of the novel image in terms of the prototypes in the training set. Since, the shape and the texture vectors of the prototypes define two completely different linear subspaces for the object class and may or may not be independent of each other, an important caveat involved here is the combined estimation of both the shape and texture parameters. Equation 6 is the basic equation that describes the mathematical formulation of the system. Further constraints may be employed, depending upon the modeling of the parameters.

4 Deformable Shape Modeling

Shape modeling in the system is done by using *finite element models* (FEM)[36, 38]. The advantage of finite element models is their ability to enforce *apriori* constraints on smoothness and amount of deformation, which in general is not possible in statistically based or optical flow based methods. FEM is a numerical approach for *modal analysis*. Modal analysis is a method for identifying the inherent dynamic characteristics of a linear system in terms of its natural frequencies, mode shapes and damping ratios. The underlying idea of modal analysis is to represent the vibration responses of a system as a linear combination of a set of simple harmonic motions[1]. This idea is similar to *Fourier analysis* of waveforms.

FEM can be used for describing non-rigid deformations of an elastic body. In this formulation, an object is modeled as a sheet of rubber which can freely deform. The surface of the object is interpolated by Galerkin method. A set of polynomial functions are defined that relate the displacement of a single point to the relative displacements of other points. Hence all the points can be expressed in terms of the interpolation functions as below:

$$\mathbf{u}(\mathbf{x}) = \mathbf{H}(\mathbf{x})\mathbf{U} \tag{7}$$

where $\mathbf{H}$ is the set of interpolation functions, $\mathbf{x}$ is a vector of all the data points and $\mathbf{U}$ is the vector of displacement components at each feature point. The strains produced at each feature point due to the displacement are obtained as a combination of the element strains associated with the feature points:

$$\epsilon(\mathbf{x}) = \mathbf{B}(\mathbf{x})\mathbf{U} \tag{8}$$

where $\mathbf{B}$ is the strain matrix and ϵ is a vector of strains produced at the point under consideration. The problem of modal displacements is then solved as a dynamic equilibrium equation:

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{D}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{R} \tag{9}$$

where $\mathbf{M}$, $\mathbf{D}$ and $\mathbf{K}$ are the mass, damping and stiffness matrices and $\mathbf{R}$ is the load matrix. The reader is directed to [38] for detailed derivation of all the mentioned matrices.

The non-rigid deformations are then expressed in an orthogonal system where the basis is defined as the set of orthonormalized eigenvectors of $\mathbf{M}^{-1}\mathbf{K}$. Given that $\mathbf{x}$ is the set of all feature points, the locations of the new feature points is given as follows:

$$\mathbf{x}' = \bar{\mathbf{x}} + \sum_{i=1}^{m} \phi_j \tilde{u}_j \tag{10}$$

where $\bar{\mathbf{x}}$ is the mean displacement position, $\mathbf{x}'$ is the deformed position, $\tilde{u}_j$ is the jth mode parameter value and ϕ_j is the jth eigenvector defining the jth modal displacement. The system can be re-orthogonalized to separate the affine parameters from the modal parameters.

4.1 Active Blobs

We use *active blobs*[37] to register the prototype images. Active blobs is a non-rigid region based finite element method used for registration and tracking of deformable objects. Initial blob of a reference object is created by associating a deformable polygonal mesh with the object texture map. Registration is solved as an energy minimization problem where the shape parameters (in our case, the finite element modes) are estimated so that difference between the warped reference object and the novel object is minimized. Minimization is done by least squares approach which will be described in detail in next section.

Let I_{ref} be the reference image from which the initial blob is created. Given a novel image I_1, the goal is to obtain a set of mode parameters that will transform the reference image to match the target image over the region of the blob. The method works by estimating the change in the parameters required to minimize the sum of squared differences error between the current estimate and the target image. In the implementation, instead of warping the reference image to the target image, the target image is inverse warped to the reference image. The energy minimization problem is formulated as follows:

$$E_{image} = \frac{1}{n}\sum_{i=1}^{n} e_i^2 \tag{11}$$

$$e_i = \| I_1'(x_i, y_i) - I_{ref}(x_i, y_i) \| \tag{12}$$

where $I_1'(x_i, y_i)$ is the intensity of the pixel at location (x_i, y_i) in the inverse warped target image I_1 and $I_{ref}(x_i, y_i)$ is the intensity of the pixel at the same location in the reference image. The adverse effect of the outliers that tend to throw the minimization process out of track are handled by using a robust error norm which is a *Lorentzian influence function* ρ, given as:

$$E_{image} = \frac{1}{n}\sum_{i=1}^{n} \rho(e_i, \sigma) \tag{13}$$

$$\rho(e_i, \sigma) = \log(1 + \frac{e_i^2}{2\sigma^2}) \tag{14}$$

where σ is an optional scale parameter.

5 Minimization Techniques

This section introduces two minimization techniques namely, *difference decomposition* and *Levenberg Marquardt*, that were used to register prototype images by active blobs and fit the model to novel images. While the former is a locally-linear approach, the latter is a non-linear approach of minimization. Model fitting will be described in the next section.

5.1 Difference Decomposition Method

Minimization of certain linear least squares problems can be accomplished via *difference decomposition*[13]. The method works by estimating derivatives of the function along each parameter. The derivatives are estimated by taking difference between the warped reference image and the original reference image. The warping is done by adding a small displacement to the parameter with respect to which the derivative is to be taken. This works well under the assumption that the function to be minimized is locally linear. Let $N = \{n_0|...|n_m\}^T$ be a matrix whose rows are the displacement vectors. Let $B = \{b_0|...|b_m\}^T$ be the matrix whose rows are the difference templates corresponding to each displacement vector obtained as mentioned above:

$$b_k = I_0 - \mathcal{W}^{-1}(I_0, n_k) \tag{15}$$

$\mathcal{W}$ is the warping function. The intuition behind the approach is as follows. During the minimization phase, if the difference image between the warped reference image and the target image is similar to the difference template b_k, then the change in the warping parameters required to register the reference image with the target image is given by n_k. Thus, if the difference image can be expressed as a combination of the difference templates, then the change in the parameters required is given by the combination of the displacement vectors. The difference patterns can be generated from the reference image, and hence are precomputed. Let the current estimate of the parameters be q. Then the difference between the inverse warped target image and the reference image is given as:

$$D = I_0 - \mathcal{W}^{-1}(I_0, q) \tag{16}$$

By the linearity assumption, this difference image D can be approximated as a combination of the difference basis vectors:

$$D \approx B^T k \tag{17}$$

where k is a vector that gives the combination coefficients. Since it is an over-constrained system, the solution is obtained by least-squares approximation whereby k is obtained as:

$$k = (BB^T)^{-1} DB^T \tag{18}$$

The corresponding change in the warping parameters is computed and the parameters are updated as below:

$$\Delta q = Nk \tag{19}$$

$$q' = q + \Delta q \tag{20}$$

5.2 Levenberg Marquardt Method

Levenberg Marquardt method is a non-linear minimization technique that performs more reliably than difference decomposition as it does not make any assumptions regarding linearity of the function. The technique uses a combination of linear and non-linear approaches for updating parameters during each iteration. Smooth switching between the two approaches is accomplished by a weighting term λ. When the magnitude of λ is low, the minimization is done in a linearized fashion by *Gauss-Newton method* whereas higher magnitude of λ forces the system to be solved in quadratic fashion by using *Gradient Descent technique.*

The mathematical formulation is as follows. Given an objective function E, the parameters of which are q', the goal is to determine an instance q that minimizes the value of E. This is achieved iteratively by solving the following set of simultaneous equations:

$$(H + \lambda I)\Delta q = g \tag{21}$$

$$q' = q + \Delta q \tag{22}$$

where H, g and λ are the Hessian matrix, the gradient vector and the controlling parameter respectively. The gradient vector and the Hessian matrix are determined as follows:

$$g_k = -\frac{\partial E}{\partial q_k} \tag{23}$$

$$h_{kl} = \frac{\partial E}{\partial q_k}\frac{\partial E}{\partial q_l} \tag{24}$$

The cost of the objective function is determined with the updated parameter values q'. If the cost has decreased as compared to its previous value then the system tends to linear minimization by scaling down λ by a factor of 10. If the cost has increased then the system moves towards quadratic minimization by scaling up λ by 10. In the former case, the parameters are updated to q', whereas in the latter case, the updated parameter vector q' is discarded and we proceed with the old parameter vector q. Higher values of λ restrict parameter displacement in the error space and force the solution to move along the *steepest gradient.*

5.3 Gaussian Pyramids

It is not uncommon to find situations where the minimization solution gets trapped in local minima. This may happen when the error function is not exactly concave or the amount of change allowed in the parameters do not move the current estimate closer to the global minima. As a result the solution gradually drifts into a local trough and eventually gets trapped inside there. Such problems can be handled reliably by using a *multigrid relaxation approach*[41]. These methods work by taking advantage of multiple discretizations and smoothing of a continuous problem over a range of resolution levels. Solution to a minimization problem requires computations proportional to the spatial distance between the current estimate and the actual solution. This suggests the possibility of speedup by computing the solution over a coarse grid and then enhance it by successively refining the grid. Pyramids are one such multi-resolution technique used in image processing[35].

The pyramids used in the current implementation are called *octave pyramids* as at each level the image is halved in each dimension and subsampled. Successive reduction in the resolution and subsampling results in the loss of high frequency components in the original image. In other words, this is equivalent to filtering the image through low-pass filters whereby the image is blurred by Gaussian kernels at each level. Thus at the coarsest level, it may be assumed that all the components corresponding to the local minima are smoothened enough to be determined as possible points of solution. Hence when successive solutions are computed from the coarsest levels and propagated to the finer levels, the solution tends towards the global minima and eventually it may be expected to converge to the actual global minima.

```
Inputs:
  • Prototypes: $I_1, I_2, \ldots, I_N$.
  • Initial reference image: $I_{ref}$.
Outputs:
  • Final reference blob: $I_{ref}$.
Steps:
  • Iterate
      – For each prototype $I_i$ do
          * Register $I_{ref}$ with $I_i$.
          * $S_i$ = set of mode values required to register $I_{ref}$ with $I_i$.
          * $T_i = \mathcal{W}^{-1}(I_i, S_i)$.
        EndFor.
      – $S_{avg} = \frac{1}{N}\sum_{j=1}^{N} S_j$.
      – $T_{avg} = \frac{1}{N}\sum_{j=1}^{N} T_j$.
      – Create new reference image:
        $I'_{ref} = \mathcal{W}(T_{avg}, S_{avg})$.
      – $I_{ref} = I'_{ref}$.
  • Until $iterations \neq n$ and
    $\sum_{x,y} \|I'_{ref}(x,y) - I_{ref}(x,y)\| \geq threshold$.
```

Figure 2: Algorithm: Average Image Computation

6 Implementation

6.1 Average Image Computation

If the reference image happens to be from a group of images that are clustered together in the prototype space, then the results would be biased towards the prototypes in that cluster. Hence we use the average image which will be fairly equidistant from each of the prototypes. This average image is computed in an iterative fashion. We start with an arbitrary reference image I_{ref}. The user circles out the region of interest from which a blob is created. The basic steps for average image computation have been enumerated in Figure 2.

6.2 Training and Matching Phases

Once the *reference* image has been computed, the system needs to be trained with each of the prototype images. This is done by registering each image with the reference image. The mode values are stored as the shape vectors and the inverse warped prototype images are stored as the texture vectors. The reconstruction of a novel object is obtained by minimizing Equation 6. The computations involved in the training and matching phases vary according to the minimization method used. The basic steps are enumerated in Figures 3 and 4.

6.2.1 Difference Decomposition Method

Difference decomposition method, assumes that the prototype space is linear. The method works by precomputing the estimates of the first derivatives of the function with respect to each parameter and using them later to approximate the current difference between the estimate and the target image as a linear combination of the precomputed derivatives.

Training Phase

Assuming that the reference image is very close to the average image, we can use difference decomposition to minimize the objective function. The intuition is that the difference between the novel image and the average image should be expressible as a combination of the differences between the average image and the prototype images. Let S_{avg} and T_{avg} be the average mode and texture vectors respectively. Then in the ideal case

$$\begin{aligned} E &= \frac{1}{2}\sum_{x,y}[\mathcal{W}^{-1}(I_{avg}, S_{avg})(x,y) - T_{avg}(x,y)]^2 \\ &+ \gamma(\sum_{i=1}^{N} c_i - 1)^2 = 0 \end{aligned} \qquad (25)$$

where all the coefficients c_i and b_i have the value $\frac{1}{N}$. The second term in the equation is a constraint term that restricts the shape parameters to sum to unity in order to account for redundancies due to the inclusion of affine parameters like scale and rotation etc. in the modal parameters. In the average case, this term in the equation is equal to zero. The difference templates can be computed by varying each parameter n_i by a value δ_i such that the energy of the difference images due to each δ_i is same as E_{thresh}, where E_{thresh} is some threshold. For each energy level a positive and a negative δ_i is determined by the *bisection method*. Initially, only one entry corresponding to a particular parameter is non-zero and all others are zero. For a single energy level E_{thresh} then the basis displacement matrix looks like $N = \{n_{0_{max}}|n_{0_{min}}|\ldots|n_{m_{max}}|n_{m_{min}}\}^T$ where $n_{i_{max}}$ and $n_{i_{min}}$ are the positive and negative displacement vectors, that have non-zero entries $\delta_{i_{max}}$ and $\delta_{i_{min}}$ respectively at the ith entry and 0 elsewhere. These displacement vectors produce difference templates $B = \{b_{0_{max}}|b_{0_{min}}|\ldots|b_{m_{max}}|b_{m_{min}}\}^T$. The matrix B may be ill-conditioned due to redundancies in the shape and the texture vectors. As a result it can't be used directly for difference decomposition as described in Section 5.

In order to remove those redundancies, we transform the displacement vectors such that the resulting difference templates would be near orthogonal. This transformation

```
Inputs:
  • Reference blob: $I_{ref}$.
  • Difference decomposition basis vectors:
    $\mathbf{N} = \{n_1, n_2, \ldots, n_m\}^a$.
Outputs:
  • Prototype texture vectors:
    $\mathbf{T} = \{T_1, T_2, \ldots, T_N\}$.
  • Prototype shape vectors:
    $\mathbf{S} = \{S_1, S_2, \ldots, S_N\}$.
  • Difference image templates:
    $\mathbf{B} = \{b_1, b_2, \ldots, b_m\}^a$.
Steps:
  1. For each prototype $I_i$ do
       • $S_i$ = set of mode values required to register $I_{ref}$ with $I_i$.
       • $T_i = \mathcal{W}^{-1}(I_i,\ S_i)$.
     EndFor.
  2. $\mathbf{S} = \{S_1, S_2, \ldots, S_N\}$.
  3. $\mathbf{T} = \{T_1, T_2, \ldots, T_N\}$.
  4. If minimization == Difference decomposition
       • $S_{avg} = \frac{1}{N}\sum_{j=1}^{N} S_j$.
       • $T_{avg} = \frac{1}{N}\sum_{j=1}^{N} T_j$.
       • For each $n_i \in N$ do
           – $\mathbf{n_{shape}} = \{n_i^1, \ldots, n_i^N\}$.
           – $\mathbf{n_{texture}} = \{n_i^{N+1}, \ldots, n_i^{2N}\}$.
           – $b_i(x,y) = \mathcal{W}^{-1}(T_{avg},\ S_{avg})(x,y) - \mathcal{W}^{-1}(\mathbf{n_{texture}} \cdot \mathbf{T}, \mathbf{n_{shape}} \cdot \mathbf{S})(x,y)$.
         EndFor.
       • $\mathbf{B} = \{b_1, b_2, \ldots, b_m\}$.
       • Save T, S and B.
     EndIf
  5. If minimization == Levenberg Marquardt then
       • Save T, S.
     EndIf
```

[a] required only for Difference Decomposition.

Figure 3: Algorithm: Training Phase

is obtained by using *singular value decomposition* (SVD) as follows.

$$B = U\Sigma V^T \tag{26}$$

$$N^* = N(B^T U_t) \tag{27}$$

where U and V are left and right singular vectors respectively and σ is the matrix of singular values of B; U_t is the truncated left singular vectors' matrix [42, 3]. The last equation is obtained as the result of linearity assumption. If $rank(B) = t$ and $R(B)$ and $N(B)$ are the range and null spaces of B then

$$R(B) \equiv span\{u_1, \ldots, u_t\} \tag{28}$$

$$N(B) \equiv span\{v_{t+1}, \ldots, v_n\} \tag{29}$$

N^* is the new set of displacement vectors obtained. Note that N^* will now have fewer displacement vectors as compared to N because of truncation and will not be sparse anymore. New difference templates $B^* = \{b_0{}^*| \ldots |b_k{}^*\}^T$ are determined corresponding to the displacement vectors in N^*. k is the point where the singular values were truncated. This new B^* matrix is now used in place of the B matrix in difference decomposition.

Matching Phase

Let the mixing parameters for the shape and the texture vectors of the prototypes be represented by $\mathbf{c} = [c_1, \ldots, c_N]$ and $\mathbf{b} = [b_1, \ldots, b_N]$. Given a novel image I_{novel}, the mixing parameters $q = [\mathbf{c}|\mathbf{b}]$ are obtained by projecting the difference image D onto the difference decomposition basis, where D is obtained as:

$$D(x,y) = \mathcal{W}^{-1}(I_{novel}, \mathbf{c} \cdot \mathbf{S})(x,y) - (\mathbf{b} \cdot \mathbf{T})(x,y) \tag{30}$$

The change in the mixing parameters k is obtained in the least squares framework as follows:

$$E = (B^{*T}k - D)^2 + \gamma(W(q + N^*k) - 1)^2 \tag{31}$$

where the second term is the constraint term expressed in matrix notation. W is an weighting vector which contains 1's corresponding to the shape terms and 0's for all texture terms. Differentiating, both sides we get

$$\begin{aligned} \frac{\partial E}{\partial k} &= 2 * [B^*B^{*T}k - DB^{*T} + \gamma N^*W^TWq \\ &- \gamma N^*W^T + \gamma N^*W^TWN^{*T}k] \end{aligned} \tag{32}$$

Rearranging the terms and evaluating to zero we get,

$$\begin{aligned} k &= [B^*B^{*T} + \gamma N^*W^TWN^{*T}]^{-1}[DB^{*T} \\ &\quad -\gamma N^*W^TWq + \gamma N^*W^T] \end{aligned} \tag{33}$$

$$\Delta q = N^*k \tag{34}$$

$$\Delta q = [\Delta \mathbf{c}|\Delta \mathbf{b}] \tag{35}$$

The mixing parameters are then updated to get the new mixing parameters and this process is iterated until they stop changing or a fixed number of iterations are complete.

$$q^{'} = q + \Delta q \qquad (36)$$

The final reconstruction has the shape of the deformed blob with mode values obtained as combination of the mode values of all the prototypes as determined by **c** parameters and the texture as determined by the **b** parameters.

6.2.2 Levenberg Marquardt Method

The training phase for this method typically involves the registration of the prototype images and determination of the shape and the texture vectors for each prototype. No precomputations are required for this method.

Matching Phase

The matching phase for a novel image involves the minimization of the error function given above. The minimization is performed by computing the first and second derivatives of the objective function and equating them according to the first approximation principle for derivatives. For simplicity, we use forward warping instead of inverse warping for this method of minimization. Hence the objective function along with its required derivatives is given as follows:

$$\begin{aligned} E(\mathbf{b},\mathbf{c}) &= \frac{1}{2}\sum_{x,y}[I_{novel}(x,y) - \mathcal{W}(\mathbf{b}\cdot\mathbf{T},\mathbf{c}\cdot\mathbf{S})(x,y)]^2 \\ &\quad +\gamma(\sum_{k=1}^{N} c_k - 1)^2 \qquad (37) \\ \frac{\partial E}{\partial b_k} &= \sum_{x,y}[I_{novel}(x,y) - \mathcal{W}(\mathbf{b}\cdot\mathbf{T},\mathbf{c}\cdot\mathbf{S})(x,y)] \\ &\quad [-\mathcal{W}(T_k,\mathbf{c}\cdot\mathbf{S})(x,y)] \qquad (38) \\ \frac{\partial E}{\partial c_k} &= \sum_{x,y}[I_{novel}(x,y) - \mathcal{W}(\mathbf{b}\cdot\mathbf{T},\mathbf{c}\cdot\mathbf{S})(x,y)] \\ &\quad [\frac{\partial\mathcal{W}(\mathbf{b}\cdot\mathbf{T},\mathbf{c}\cdot\mathbf{S})}{\partial c_k}(x,y)] \\ &\quad +2\gamma(\sum_{k=1}^{N} c_k - 1) \qquad (39) \end{aligned}$$

$$\begin{aligned} \frac{\partial\mathcal{W}(\mathbf{b}\cdot\mathbf{T},\mathbf{c}\cdot\mathbf{S})}{\partial c_k}(x,y) &= [\mathcal{W}(\mathbf{b}\cdot\mathbf{T},(\mathbf{c}+\Delta\mathbf{c})\cdot\mathbf{S})(x,y) \\ &- \mathcal{W}(\mathbf{b}\cdot\mathbf{T},\mathbf{c}\cdot\mathbf{S})(x,y)]/\Delta \quad (40) \end{aligned}$$

$$\Delta c_k = [0, \ldots, k - 1 times, \Delta, 0, \ldots, 0] \qquad (41)$$

```
Inputs:
  • Prototype shape and texture vectors:
    $\mathbf{S} = \{S_1, \ldots, S_N\}$ and $\mathbf{T} = \{T_1, \ldots, T_N\}$.
  • Novel Image: $I_{novel}$.
  • Basis vectors and difference templates[a]:
    $\mathbf{N} = \{n_1, \ldots, n_m\}$ and $\mathbf{B} = \{b_1, \ldots, b_m\}$.
Outputs:
  • Combination parameters: c and b.
  • Reconstructed novel image: $I_{reconstruction}$.
Steps:
  • $\mathbf{c} = \mathbf{b} = \{\frac{1}{N}, \ldots, \frac{1}{N}\}$; $\mathbf{q} = \{\mathbf{c}|\mathbf{b}\}$.
  • If minimization == Difference decomposition
      – Iterate
          ∗ Compute $D$.
          ∗ $\Delta q = N([BB^T]^{-1}DB^T)$.
          ∗ $q = q + \Delta q$.
          ∗ Compute $E(\mathbf{c}, \mathbf{b})$.
      – Until $iterations \neq n$ and
        $\|E\| \geq threshold$.
    EndIf
  • If minimization == Levenberg Marquardt
      – Iterate
          ∗ $\Delta q = (H + \lambda I)^{-1} g$.
          ∗ $q' = q + \Delta q$.
          ∗ $\mathbf{c}' = \{q'^1, \ldots, q'^N\}$.
          ∗ $\mathbf{b}' = \{q'^{N+1}, \ldots, q'^{2N}\}$.
          ∗ Compute $E(\mathbf{c}', \mathbf{b}')$.
          ∗ If $\|E\|$ decreased then
              · $\mathbf{q} = \mathbf{q}'$; $\lambda = \lambda/10$.
            else
              · $\lambda = \lambda * 10$.
            EndIf
      – Until $iterations \neq n$ and
        $\|E\| \geq threshold$.
    EndIf
  • $\mathbf{c} = \{q^1, \ldots, q^N\}$; $\mathbf{b} = \{q^{N+1}, \ldots, q^{2N}\}$.
  • $I_{reconstruction} = \mathcal{W}(\mathbf{b}\cdot\mathbf{T},\ \mathbf{c}\cdot\mathbf{S})$.
```

[a] required only for Difference Decomposition.

Figure 4: Algorithm: Matching Phase

The second derivatives of the given function are approximated as below:

$$\frac{\partial^2 E}{\partial m_i \partial m_j} = \frac{\partial E}{\partial m_i}\frac{\partial E}{\partial m_j} \tag{42}$$

where $m_k = c_k$ or b_k. These derivatives so obtained are then used to define the gradient vector and the Hessian matrix and the system of equations are solved as described in Section 5.2. The parameter vector q is defined as a composite vector of the shape and the appearance parameters:

$$q = [\mathbf{c}|\mathbf{b}] \tag{43}$$

This process is iterated until the final error magnitude drops below a given threshold or a fixed number of iterations are completed. The λ in the equation above, acts as a time-varying control parameter that forces the solution to follow the steepest gradient in order to converge to the minimum.

7 Feasibility Study

The system was implemented with both the strategies described. The system was tested with two types of datasets, namely face images and sequences of heart images, and was tested with some images from the training set as well as some novel images that were not present in the training set. It was observed that the implementation worked well with Levenberg Marquardt method whereas it failed to produce satisfactory results with difference decomposition method. The possible explanations for the observed behavior and further extensions are discussed in Section 8.

7.1 Test Set 1: Face Images

Figure 5: Average face image

The code for registration of images was taken from *Active Blobs* which is available on the internet[2]. The prototype set comprises of random face images drawn from the MIT database (see Figure 6). Several novel face images, which were not present in the training set, were tested. All of those could be reconstructed in the combination of parameters paradigm described earlier. The images are of dimension 128x128 pixels. The size of the faces inside the images was typically around 64x64 pixels. The implementation makes extensive use of the graphics hardware for texture mapping and bilinear interpolation. Currently

Figure 6: Prototype face images (#prototypes = 100)

[2] http://www.cs.bu.edu/groups/ivc/

the reconstruction of a novel face image takes around 8 minutes on a R5000 SGI O2, 180 MHz machine. Majority of time is spent in combining the prototype images at each iteration for the reconstruction of novel image. The texture vectors for the prototype images comprise of the whole texture. Significant speedup is possible by dimensionality reduction. In future, we intend to evaluate the system with *dimensionally reduced* prototype texture vectors, where we will use coefficients obtained by projecting prototype images into the eigenspace instead of textures. We expect the performance of the system would improve as we will have to combine less number of eigen-images for reconstruction. We can further reduce the computation time by doing minimization on only one color channel. Various other alternatives are also currently being explored to improve the computation time (see section 8). The following paragraphs explain the working of the system using Levenberg-Marquardt method.

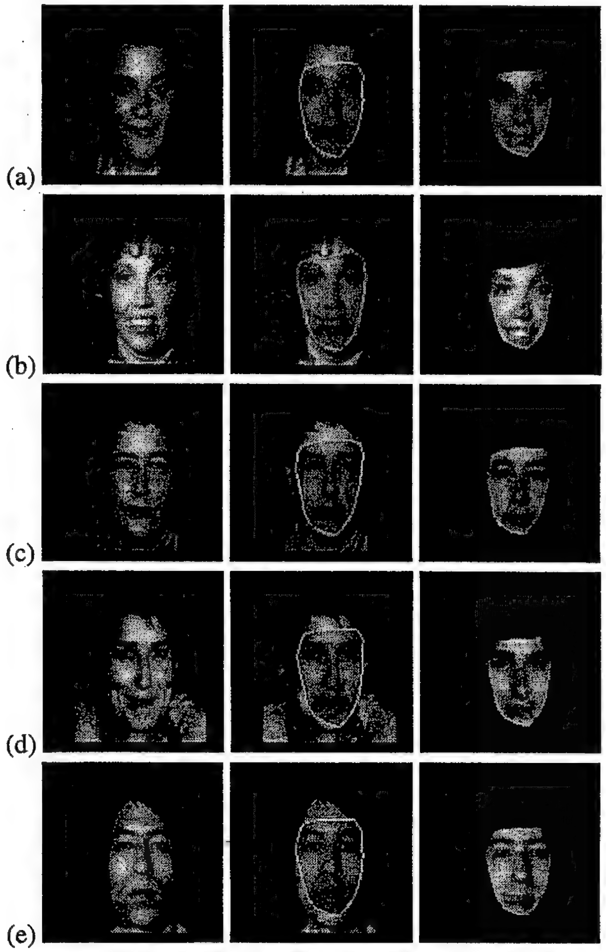

Figure 7: Reconstructed novel face images. Left: input novel image; Middle: average image registration; Right: reconstruction

The user starts with the *average image computation* phase. In this phase the user circles the region of interest (in the current experiment, outlines the face in the image) and a blob is created from the encircled region. This blob is then used to register other face images. For each prototype image, the user moves the blob and places it over the face. Then the blob is allowed to automatically deform and register with the face underneath it. Shape and texture parameters are recovered. Since all the face images in the MIT database have been placed such that the positions of their eyes coincide, we can automate this process by placing the blob at a designated position in the image, rather than having the user place it. Alternately, we can use a face detector for initial placement of the blob, but then it would handle only face images. The outcome of this stage is the average blob which is then saved for future use.

In the *training phase*, the average blob is loaded and the user trains the system by registering the prototype face images. This phase can also be automated as described in the previous paragraph.

The *matching phase* involves the reconstruction of the novel image specified by the deformable model. Final results are presented in the form of combination coefficients for shape and texture parameters. We propose to evaluate the quality and reliability of reconstruction in terms of error residuals and correlation soon.

Figure 8: Prototype heart images (#prototypes = 20)

In this experiment, we used a test set comprising of 100 prototype images (Figure 6). The prototypes comprised of 75 images of males with mustaches and 25 images of females. Figure 7 displays the reconstructions of some novel images. The main points that were tested here are: (1) the algorithm is able to reconstruct novel faces by combining prototype faces; (2) the algorithm is capable of handling significant variations ("gender", in this case); (3) the linear combinations paradigm can be exploited to generate new images, that have not been seen earlier (male images without mustaches, in this case). The top two rows in Figure 7 display the reconstructions of two novel images of females, not present in the training set. The last three rows display the reconstruction of male images without any mustaches. Note that the training set did not contain any male prototype images without mustaches. The algorithm was still able to use the linear combinations paradigm to appropriately combine male and female prototype images to reconstruct those novel images.

7.2 Test Set 2: Sequences of Heart Images

In order to evaluate the generality of the approach described, we tested the system with images other than faces. The second test set included sequences of heart pumping taken from the MIT heart database. Since there were only 38 images, we included all the odd numbered images into the training set and used the even numbered images as novel images. The images used for training are given in Figure 8. The average image for this sequence of images and reconstruction of some novel images are given in Figures 9 and 10. The estimated shape and texture parameters can be used for various medical applications.

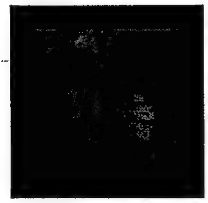

Figure 9: Average heart image

8 Discussion

In this section, we analyze the various observations made during the preliminary experiments.

8.1 Difference Decomposition Method

Despite being an elegant real time minimization method, difference decomposition fails to produce satisfactory results in the current framework. The reason for this is the breakdown of the local linearity assumption. In general, for applications like tracking (as in Active Blobs), there is an underlying assumption that the motion between two consecutive frames follow *image space constancy* which forbids extreme changes between two consecutive frames. This assumption may not hold in the current methodology as it may be the case that several prototypes are not similar to the average image, hence the energy surface cannot be assumed to be smooth. Possible ways of addressing this problem are given as follows:

- *Recompute the difference images at each iteration*
  Although this may help in driving the system to the actual solution, still the overheads involved pull down the very importance of the method as performance will become extremely slow.

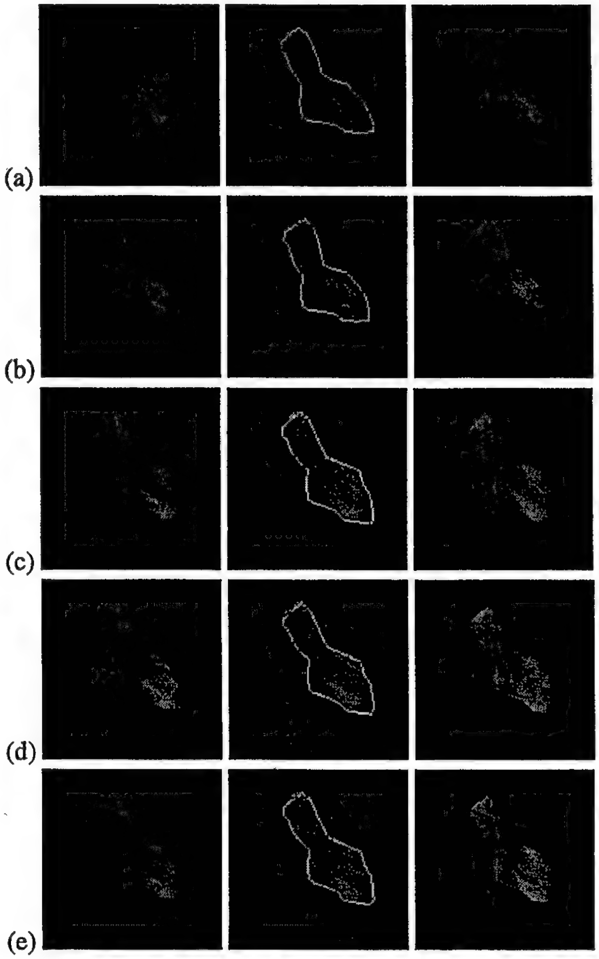

Figure 10: Reconstructed novel heart images. Left: input novel image; Middle: average image registration; Right: reconstruction

- *Time-varying Jacobian matrix*
  Hager and Belhumeur[14] had derived a method similar to difference decomposition, where the *Jacobian matrix* is time-varying rather than being completely precomputed as proposed by Gleicher[13]. This suggests that it may be possible to separate out the two varying components (shape and texture) and use an iterative technique where only one of the parameters is modified at a time as Hager and Belhumeur do by using an *iteratively reweighted least squares method.*

- *Active Appearance Models*
  Cootes, Edwards and Taylor[6] have used a method similar to difference decomposition. However, they control the parameter update step by appropriately scaling the amount of change in parameters at each iteration. This is done by adding the following steps after Equation 20:

  1. $k = 1$.
  2. Compute the energy E with updated parameters.

```
3. If magnitude of $E$ has increased from its previous value, then
4.     Set $k = k/2$.
5.     Set $q' = q + k * \Delta q$, goto 1.
6. Else
7.     Set $q = q'$.
```

This in theory restricts the system from diverging by permitting small displacements such that the energy of the system decreases monotonically. However, this technique is highly susceptible to get trapped in the nearest local minima in the solution space and hence is unsuitable for image reconstruction. Applications have been shown for tracking where a rough fit is sufficient to track and moreover, it is possible that iterations over several frames may pull out the solution from the local minima and drive it towards the global minima.

8.2 Levenberg Marquardt Method

Though Levenberg Marquardt, being a non-linear quadratic minimization method addressed the problem of linearity, it is plagued by large computation times required for the Hessian matrix. The major bottleneck is the number of floating point multiplications involved which is $O(n^2m^2)$ where each image has $O(n^2)$ pixels and there are $O(m)$ prototype images. Possible ways of improving the current status may involve the following:

- *Minimization at random pixels*
  Instead of computing the Hessian matrix over all the pixels in the images, we can choose random pixels at each iteration. This, in theory, tries to simulate the *stochastic gradient method*[42], but would be more efficient as it would converge to the solution faster by taking larger step sizes (implicitly controlled by λ) at each iteration, if the error function happens to be purely quadratic.

- *Number of prototypes used*
  The contribution of the number of prototype images is at least quadratic with respect to the amount of computations performed. A significant improvement may be obtained by selecting "good" prototypes and excluding redundant prototypes. This issue is addressed in the next section.

8.3 Prototype Selection

Currently, a set of prototypes is chosen randomly and the training is done on this set. If three prototypes are thought to lie on a line in the prototype space, then any number of extra prototypes on the same line are redundant and hence should be singled out. Though different statistical methods like k-means clustering, hierarchical clustering, Bayes classifier etc. can be used, a normal tradeoff involved is that typical pattern recognition methods require large training data sets which are diverse enough to characterize the whole object class[10, 36]. Some possible techniques for manual or automatic selection of prototypes and their drawbacks are as follows:

- *Manual selection*
  Human beings, may not be good judges of variations across different object classes, hence virtually make this approach impossible.

- *Approximation by MDL*
  An useful way of estimating the good prototypes is the *minimum description length method,* also known as MDL. But the major drawback of this method is the implicit requirement of trying out all possible combinations of the prototypes and finally select a subset that was best in spanning the whole set of the available prototypes.

- *Unsupervised techniques*
  Use an unsupervised technique for determining similarity between various prototypes and determine a hierarchical grouping structure. One such technique is *agglomerative hierarchical clustering*[10, 16]. Another way of grouping prototypes in an hierarchical fashion was suggested by Hill and Taylor[15]. But this method does not suggest a way of removing the redundant prototypes, which may be possible by pruning.

- *Naive Approaches*
  Create a similarity metric for measuring the similarity between two prototypes. Initially we train the system with all the prototypes and then determine the similarity between each of the prototypes in the dataset. We then drop redundant prototypes, retaining only one representative prototype from the group of prototypes that were grouped together as similar.

  Start with only one random prototype image in the training set. At each step try to reconstruct another prototype image. If the prototype could be reconstructed satisfactorily, then discard it else include it into the training set and re-train the system.

Currently we are implementing certain heuristics based clustering techniques that will enable us to choose "good" prototypes in future.

8.4 Active Appearance Models

The prototype spaces can be efficiently compressed using PCA in both the shape and the texture space separately. In Active Appearance Models (AAM), the dimensionality of the composite shape and texture is further reduced by PCA. This assumes that composite shape and texture spaces can be modeled as a linear space. Though this would provide a greater computational efficiency, still it is not clear how the performance of the system would be affected when some shape information is dependent on the texture information or vice versa. For instance, a change in shade in a region can occur due to local shape deformation as well as actual variation in the object appearance. AAM can learn this variation only in the texture space and might lose information associated with local shape deformation whereas the deformable model presented would learn it as a shape deformation or texture variation or both depending upon the variations exhibited by the set of prototypes.

The question for using either of the mentioned approaches depends upon the requirements of the applications for which the system is being designed. Both methods have been proved to be comparable in literature. For example, for tracking purposes where computational speed is a big factor, AAM is more appropriate. For applications like video compression by using techniques like morphing etc. intermediate sequences may easily be generated by doing straightforward interpolation of the mixing parameters. Intuitively, this seems to more closely resemble human perception[23]. In AAM, it is not evident whether simple interpolation would provide good results.

Lastly, AAM is a statistically based model. Such models can easily be confused and may not perform well if the prototypes provided have large variations whereas we can still expect an optimal solution in our formulation. This emphasizes another aspect that the our model will require much less number of prototypes for training as compared to AAM which will need much more prototypes to reliably learn the object class.

8.5 Invariants

An important point to ponder in the linear combinations paradigm is that given slight variability in the same novel image conditions, each time a different set of shape and texture coefficients may be recovered. For instance, under different illumination conditions, the combination of the prototype textures would change as a different group of prototypes would become important to express the observed changes. Similar observations can be made when the pose of the object changes. This variability of the combination parameters will render the model unsuitable for recognition purposes. Hence a set of possible invariants need to be derived from the current framework. Recently a method called *quotient image* has been proposed by Riklin-Raviv and Shashua[34], which provides a way of computing *signature images* for different face images under varying illumination. The same idea can easily be adapted in our model. Once the novel image has been reconstructed, the shape parameters can be used to warp the texture to a canonical shape and then the estimated texture can be used to obtain a signature image.

Given that the objects under consideration are Lambertian, the appearance can be defined as:

$$T_i(x,y) = \rho_i(x,y)n(x,y)^T s_j \qquad (44)$$

where $T_i(x,y)$, $\rho(x,y)$ and $n(x,y)$ are the intensity value, surface albedo and the direction of the normal at point (x,y) respectively. s_j is the direction of point light source. The signature or *quotient* image (Q) can be derived as a normalization of the albedo as follows:

$$Q(x,y) = \frac{\rho(x,y)}{\rho_{avg}(x,y)} \qquad (45)$$

Assuming that the shape parameters can be determined accurately, when warped on to the same surface, the dot product of the light source and the surface normal would be nearly constant. Also assuming that the prototype images appropriately span the varying illumination space, the signature image in the flexible model can be derived as below:

$$T_{signature}(x,y) = \frac{(\tilde{\mathbf{b}} \cdot \mathbf{T})(x,y)}{T_{avg}(x,y)} \qquad (46)$$

where $\tilde{\mathbf{b}}$ is the set of estimated texture parameters for the novel image.

8.6 Extendibility

The framework described here can easily be extended to handle parameters other than just shape and texture. Illumination modeling can be included in the current system by creating a separate basis for illumination variation[30]. If the prototype images are taken appropriately under different illumination conditions (say, with frontal light source), the illumination can be modeled separately. Illumination prototypes can be obtained by taking several images of the average/reference image under various lighting condition. These illumination prototypes can then be used to create illumination basis as used by Hager and Belhumeur[14]. The new objective function is given as below:

$$\begin{aligned} E(\mathbf{c},\mathbf{b},\beta) &= \frac{1}{2}\sum_{x,y}[\mathcal{W}(I_{novel},\mathbf{c}\cdot\mathbf{S})(x,y) - (\mathbf{b}\cdot\mathbf{T})(x,y) \\ &- (\beta\cdot\mathbf{B})(x,y)]^2 + \gamma(\sum_{k=1}^{N} c_k - 1)^2 \qquad (47) \end{aligned}$$

where $\mathbf{B}$ and β are the set of illumination bases and the combining parameters for those bases respectively.

9 Applications

The idea explored in this paper is central to the several image analysis problems. Some of the possible applications are as follows:

- **Image Registration:**
  This model can be used for registering deformable models by reconstructing the shape and the texture parameters from a set of prototypes. The mixing parameters obtained can be used for clustering similar objects in a shape+appearance database.

- **Image Analysis:**
  Apart from being able to model deformable objects, this model has the added benefit of the information contained in the texture. After registration/reconstruction the texture information can be used for different analyses.

  Facial expression recognition:
  After registering a face, the texture information can be used to classify the expression on a person's face. Given that the object space is linear, if we include images of same person with different expressions, then sufficient number of prototype images should be able to define clusters in the prototype space where prototypes with similar expressions get grouped together. The novel image would then be nearer to a particular group than other prototype groups giving us information about expression.

  MR Image registration and analysis:
  Reconstruction parameters in MR images registration might give some information whether some organ is normal in appearance or shape. Such an application was reported in [25] which relies on modal shape deformations. Use of texture information might help in determining inflammation, bleeding etc.

  Image Compression:
  The above mentioned methods may be extended to do analysis over time. Studying patterns of movement between different prototypes may provide information about the kind of action occurring. Such methods can be used for video compression. Such a technique for animation using example image sequences has been reported in the past by Poggio and Brunelli[33].

- **Morphing:**
  Given that the model parameters have been estimated, it would be interesting to see how different views can be generated. It has been shown in the past the novel views of an object can be obtained by warping the reference image to different prototypes[45]. This idea can be used to extend the work done in [39] to include appearance along with physical deformations.

  The deformable model can also be used for *morphing* between multiple images as a graphics application. This model can be used to improve the work reported in [22] and may be extended for video compression.

10 Summary

In this paper, we explored a model-based linear combinations approach for modeling objects. The methodology, implementation status, results obtained so far and possible explanations of various observed behavior were described in detail . Apart from these, the method was compared with existing active appearance model and the pros and cons were brought out. Also various ways of extending the existing framework have also been described.

References

[1] K. Bathe. *Finite Element Procedures in Engineering Analysis*. Prentice Hall, 1982.

[2] A. M. Baumberg and D. C. Hogg. An efficient method for contour tracking using active shape models. Technical Report 94.11, University of Leeds, School of Computer Studies Research Report Series, April 1994.

[3] M. W. Berry and S. T. Dumais. Using linear algebra for intelligent information retrieval. Technical Report CS-94-270, University of Tennesse-Knoxville, Computer Science Dept., December 1994.

[4] H. H. Bulthoff, S. Y. Edelman, and M. J. Tarr. How are three-dimensional objects represented in the brain? *Cerebral Cortex*, 5(3):247–260, 1995.

[5] T. F. Cootes, D. H. Cooper, C. J. Taylor, and J. Graham. Trainable method of parametric shape description. *Image and Vision Computing*, 10(5):289–294, June 1992.

[6] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *ECCV98*, 1998.

[7] T. F. Cootes and C. J. Taylor. Combining point distribution models with shape models based on finite element analysis. *Image and Vision Computing*, 13(5):403–409, June 1995.

[8] M. Covell. Spanning the gap between motion estimation and morphing. *IEEE International Conference on Acoustics, Speech and Signal Processing*, April 1994.

[9] M. Covell. Eigen-points: Control-point location using principal component analyses. *IEEE Int'l Conference on Automatic Face and Gesture Recognition*, October 1996.

[10] R. O. Duda and P. E. Hart. *Pattern Recognition and Scene Analysis*. John Wiley, New York, 1973.

[11] S. Y. Edelman and H. H. Bulthoff. Viewpoint-specific representations in three dimensional object recognition. Technical Report A.I.Memo 1239, MIT, 1990.

[12] T. Fromherz, P. Stucki, and M. Bischel. A survey of face recognition. Technical Report MML Technical Report No. 97.01, University of Zurich, 1997.

[13] M. Gleicher. Projective registration with difference decomposition. In *Proc. CVPR*, 1997.

[14] G. D. Hager and P. N. Belhemeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 20(10), October 1998.

[15] A. Hill and C. J. Taylor. Automatic landmark generation for point distribution models. *British Medical Vision Conference*, 1994.

[16] A. K. Jain and R. C. Dubes. *Algorithms for Clustering Data*. Prentice Hall, Englewood Cliffs, N. J., 1988.

[17] M. Jones and T. Poggio. Model-based matching of line drawings by linear combinations of prototypes. Technical Report AI Memo No. 1559, CBIP Paper No. 128, M.I.T. AI Lab. and CBIP Whitaker College, December 1995.

[18] M. Jones and T. Poggio. Model-based matching by linear combinations of prototypes. Technical Report AI Memo No. 1583, CBIP Paper No. 139, M.I.T. AI Lab. and CBIP Whitaker College, November 1996.

[19] M. Jones and T. Poggio. Multidimensional morphable models. In *International Conference of Computer Vision*, Bombay, India, January 1998.

[20] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, pages 321–331, 1988.

[21] A. Lanitis, C. J. Taylor, and T. F. Cootes. Automatic face identification system using flexible appearance models. *Image and Video Computing*, 13(5):393–402, June 1995.

[22] S. Lee, G. Wolberg, and S. Y. Shin. Polymorph: Morphing among multiple images. *IEEE Computer Graphics and Applications*, pages 58–71, January/February 1998.

[23] N. Logothetis, J. Pauls, and T. Poggio. Viewer-centered object recognition in monkeys. Technical Report AI Memo No. 1473, CBCL Paper No. 95, M.I.T. AI Lab. and CBIP Whitaker College, April 1994.

[24] N. Logothetis, J. Pauls, and T. Poggio. Shape representation in the inferior temporal cortex of monkeys. *Current Biology*, 5(5):552–563, 1995.

[25] J. Martin, A. Pentland, and S. Sclaroff. Characterization of neuropathological shape deformations. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 20(2), february 1998.

[26] B. Moghaddam, W. Wahid, and A. Pentland. Beyond eigenfaces: Probabilistic matching for face recognition. *Int'l conference on Automatic Face and Gesture Recognition*, April 1998.

[27] H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 15:5–24, 1995.

[28] C. Nastar, B. Moghaddam, and A. Pentland. Generalized image matching: Statistical learning of physically-based deformations. In *ECCV96*, Cambridge, England, April 1996.

[29] S. Nayar and T. Poggio, editors. *Early Visual Learning*. Oxford University Press, 1996.

[30] J. Nimeroff, E. Simoncelli, and J. Dorsey. Efficient re-rendering of naturally illuminated environments. In *Proceedings of the Fifth Annual Eurographics Symposium on Rendering*, Darmstadt, Germany, June 1994.

[31] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[32] A. Pentland and S. Sclaroff. Closed-form solutions for physically based modeling and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):715–729, July 1991.

[33] T. Poggio and R. Brunelli. A novel approach to graphics. Technical Report AI Memo No. 1354, CBIP Paper No. 71, M.I.T. AI Lab. and CBIP Whitaker College, February 1992.

[34] T. Riklin-Raviv and A. Shashua. The quotient image: Class based re-rendering and recognition with varying illuminations. Technical Report TR-99-1, The Hebrew University of Jerusalem, Computer Science, January 1999.

[35] A. Rosenfeld, editor. *Multiresolution Image Processing and Analysis*. Springer-Verlag, New York, 1984.

[36] S. Sclaroff. Deformable prototypes for encoding shape categories in image databases. *Pattern Recognition*, 30(4), April 1997.

[37] S. Sclaroff and J. Isidoro. Active blobs. In *International Conference of Computer Vision*, Bombay, India, 1998.

[38] S. Sclaroff and A. Pentland. Modal matching for correspondence and recognition. Technical Report 201, M.I.T. Media Laboratory Perceptual Computing Section, May 1993.

[39] S. Sclaroff and A. Pentland. Physically-based combinations of views: representing rigid and nonrigid motion. In *Proc. IEEE Workshop on Nonrigid and Articulate Motion*, pages 158–164, Austin, TX, November 1994.

[40] P. Sinha. *Perceiving and recognizing 3D forms*. PhD thesis, Massacheussetts Institute of Technology, 1995.

[41] D. Terzopoulos. Image analysis using multigrid relaxation methods. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 8(2):129–139, 1986.

[42] S. Teukolsky, W. Press, B. Flannery, and W. Vetterling. *Numerical Recipes in C*. Cambridge University Press, UK, 1988.

[43] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):72–86, 1991.

[44] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 13(10), October 1991.

[45] T. Vetter. Synthesis of novel views from a single face image. Technical Report 26, Max-Planck Institute for biological cybernetics, February 1996.

[46] T. Vetter and T. Poggio. Linear object classes and image synthesis from a single example image. Technical Report AI Memo No. 1531, CBIP Paper No. 119, M.I.T. AI Lab. and CBIP Whitaker College, March 1995.

OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE August 1999	3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE

Combinations of Deformable Shape Prototypes.

5. FUNDING NUMBERS

G N00014-96-1-0661

6. AUTHOR(S)

Saratendu Sethi and Stan Sclaroff

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Computer Science Department
Boston University
111 Cummington Street
Boston, MA 02215

8. PERFORMING ORGANIZATION REPORT NUMBER

sclaroff-ONR-TR99-007

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Department of the Navy
Office of Naval Research
Ballston Centre Tower One
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

WE present a model-based technique for encoding non-rigid object classes in terms of object prototypes. Objects from the same class can be parameterized by identifying shape and appearance invariants of the class to devise low-level representations. The approach presented here creates a flexible model for an object class from a set of prototypes. This model is them used to estimate the parameters of low-level representation of novel objects as combinations of the prototype parameters. Variations in the object shape are modeled as non-rigid deformations. Appearance variations are modeled as intensity variations. Given sufficient number of prototypes that exhibit appropriate in-class variations, the shape and the texture vectors define a linear prototype subspace that spans the object class. The strengths of this technique lie in the combined estimation of both shape and appearance parameters. This is in contrast with the previous approaches where shape and appearance parameters were estimated separately.

14. SUBJECT TERMS

Non-rigid shape description and recognition; image and video database modeling, content-based retrieval

15. NUMBER OF PAGES

16

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
unclassified	unclassified	unclassified	UL

ENCL (1)